



# **Introducción a ASP.NET de *Microsoft*®**

**Curso de Extensión Universitaria**

**Cod. 1830.036 - ARQUITECTURA WEB EN  
APLICACIONES EMPRESARIALES BASADAS  
EN TECNOLOGÍA JAVA/J2EE**

***Director: Daniel Fernández Lanvín***

**Marzo-Abril de 2004**

**Aquilino Adolfo Juan Fuente**



# Temario

- **Resumen**
- **Introducción**
- **Arquitectura**
- **Marco de páginas**
- **Controles de servidor**
- **Administración de estado**
- **Almacenamiento en caché**
- **Enlace de datos**
- **Seguridad**
- **Configuración**
- **Referencias**



# Resumen

- **En este curso se verá...**
  - Introducción a la arquitectura ASP.NET
  - Introducción a la creación de páginas en ASP.NET
  - Introducción a seguridad y otros elementos importantes de ASP.NET
  - Links importantes para ampliar información

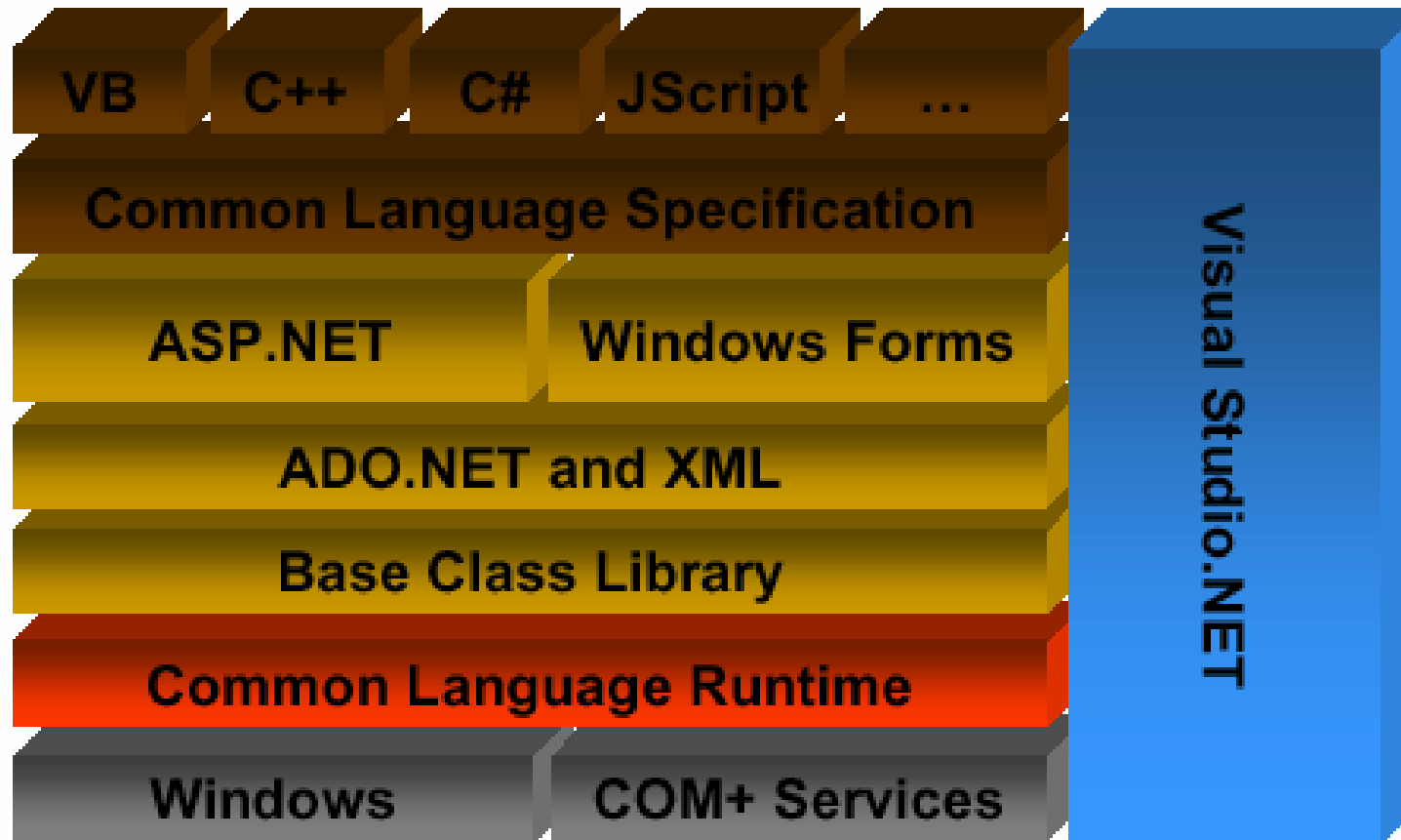


# Introducción

- **ASP.NET es una de las piezas esenciales de Microsoft .NET Framework y proporciona la infraestructura para aplicaciones .NET Web dinámicas fácilmente desarrolladas.**
- **ASP.NET no es sólo el sucesor de páginas Active Server (ASP) de Microsoft, es una plataforma unificada de desarrollo Web que proporciona a los desarrolladores los servicios necesarios para generar aplicaciones Web de empresa.**
- **ASP.NET incluye grandes mejoras con respecto a ASP e incluye muchas características nuevas.**
- **Para obtener una introducción breve a ASP.NET, se puede consultar el tema siguiente de la documentación del Kit de desarrollo de software (SDK) de Microsoft .NET Framework:**
  - Introduction to ASP.NET
    - <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguidnf/html/cpconintroductiontoasp.asp>
- **Otro buen lugar para empezar es el tutorial ASP.NET QuickStart, que se puede encontrar en el siguiente sitio Web de Microsoft GotDotNet o en el siguiente recurso compartido de su equipo local:**
  - <http://www.getdotnet.com/quickstart/aspplus>
  - <http://localhost/quickstart/aspplus>



# Arquitectura (.NET)





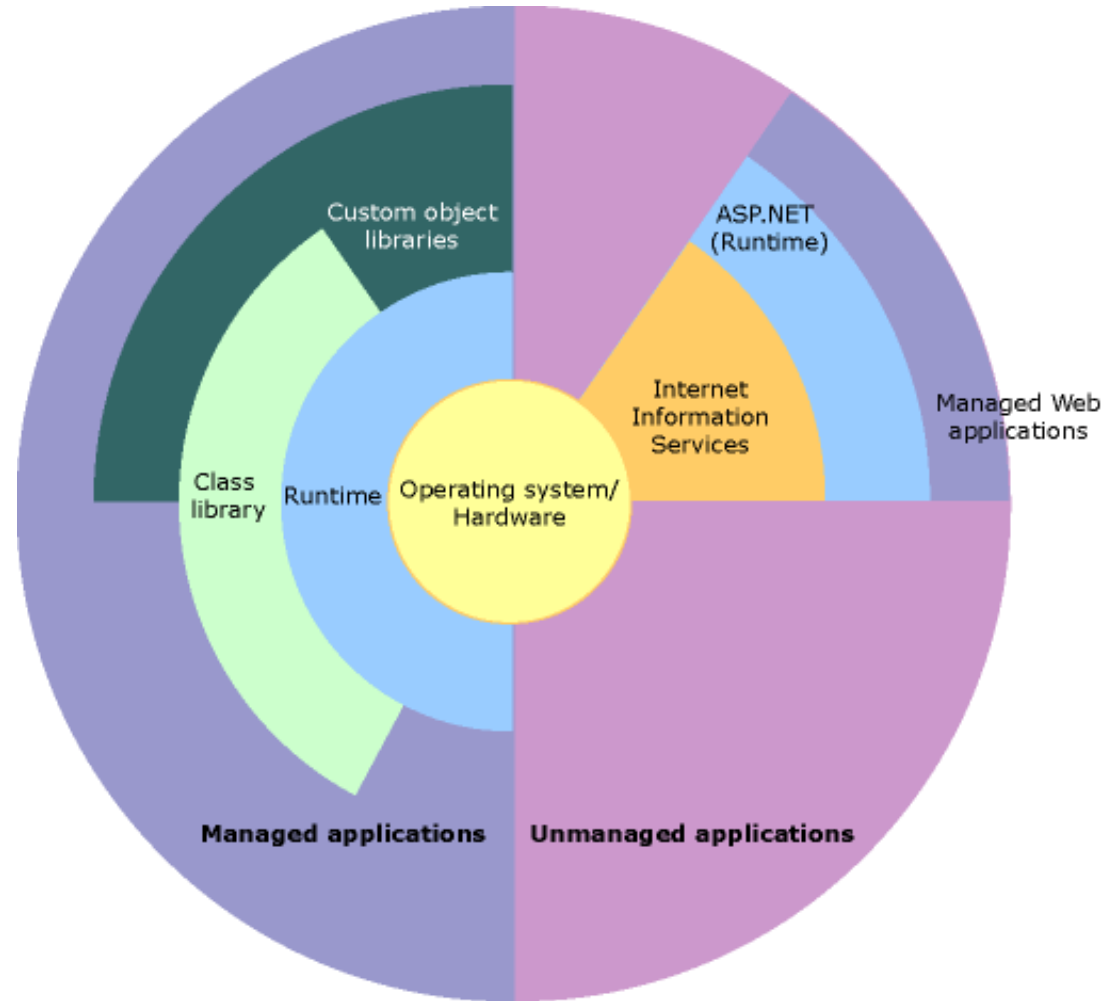
# Arquitectura (.NET)

- **.NET ofrece independencia de lenguaje e interoperabilidad entre lenguajes. Existen ya 22 compiladores (ADA, RPG, COBOL, Fortran, Eiffel, PERL, Component Pascal, C#, J#, Visual.NET).**
- **El código (sin importar el lenguaje en que esté) es traducido al MS Intermediate Language (MSIL o IL)**
  - analogía con Java bytecode.
- **La traducción del código IL a lenguaje de máquina es hecha por el Common Language RunTime (CLR)**
  - analogía con JRE.
- **Es necesario tener instalado el .NET Framework para poder correr los aplicativos desarrollados para ésta tecnología - tanto si es sobre plataforma Windows (W98 a XP) u otra. Hoy ya existen implementaciones del Framework también para Linux y FreeBSD.**



# Arquitectura (.NET)

- **Tiene 2 componentes principales**
  - CLR
  - .NET Framework class library
- **Tipos de Aplicaciones**
  - Administradas
  - No administradas





# Arquitectura (.NET)

- **Intermediate Language**
  - También conocido como “managed code”.
  - Generado por cualquier compilador con soporte de “.NET runtime”.
  - Provee la misma forma de representar los datos, por lo que permite tener herencia multilinguaje.
  - Al compilar no solo se genera el IL, sino también la metadata que permite interpretar ese “managed code”.



# Arquitectura (.NET)

- **CLR**

- “an agent that manages code at execution time, providing core services such as memory management, thread management, and remoting, while also enforcing strict type safety and other forms of code accuracy that ensure security and robustness”
- Provee características no incluidas en Windows DNA:
  - Manejo automático de “Garbage Collector”.
  - Manejo de excepciones.
  - Herencia entre lenguajes.
  - Debugging.
  - Side-by-side execution de diferentes versiones del mismo componente.



# Arquitectura (.NET)

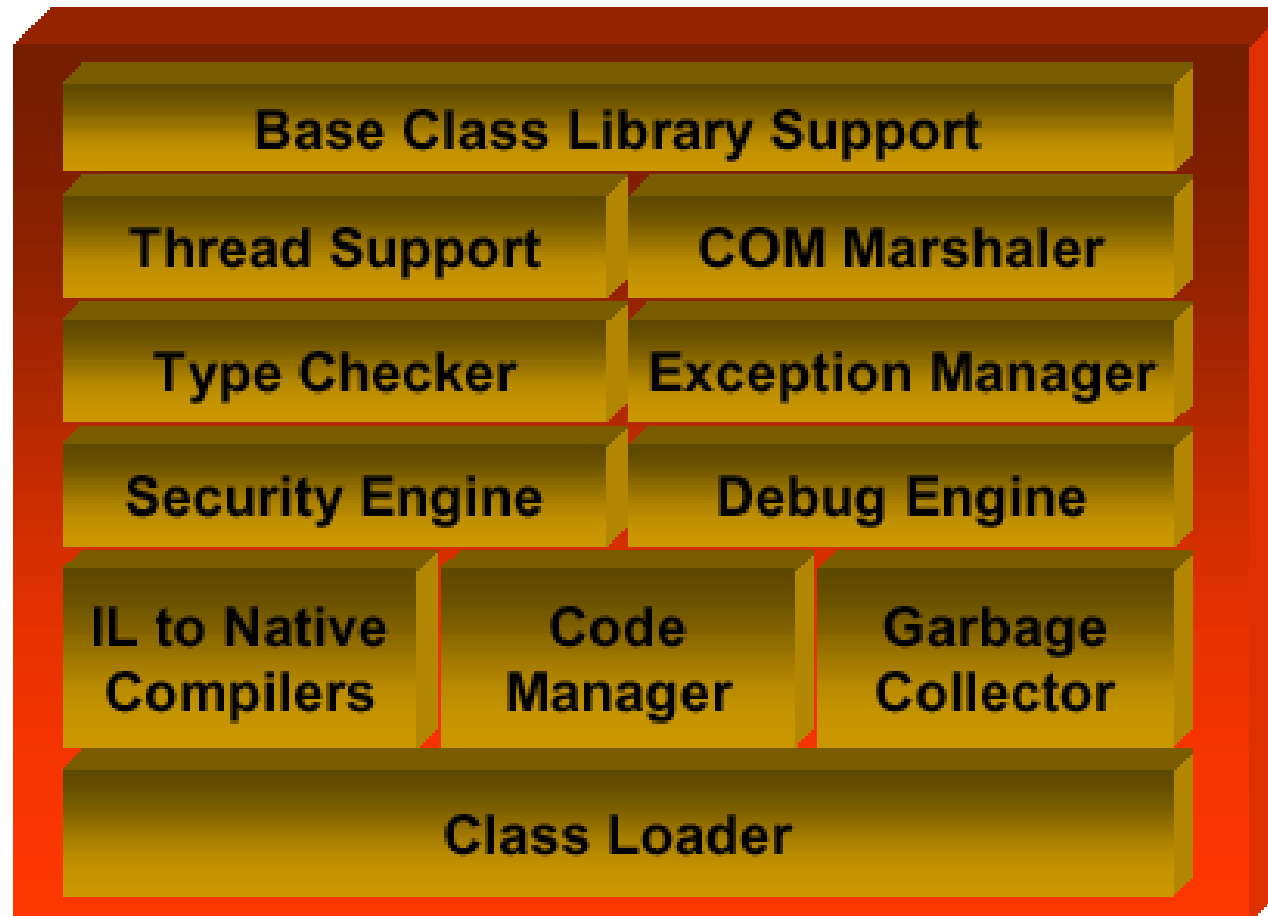
- **CLR**

- Es el motor que corre y administra la ejecución del código IL.
- Realiza la compilación JIT que traduce el “managed code” en código nativo sobre la arquitectura de hardware que esté corriendo. Esto le permite estar posicionado en forma ideal para controlar la seguridad, algo que no puede hacerse con código ya compilado en lenguaje nativo de antemano, ya que puede ser puesto a correr con permisos incorrectos.



# Arquitectura (.NET)

- CLR





# Arquitectura (.NET)

- **Class Library**

- Una colección orientada a objetos de tipos reutilizables que se pueden utilizar para desarrollar aplicaciones que van desde una aplicación tradicional de línea de comando o una aplicación gráfica, hasta aplicaciones basadas en las últimas tecnologías como ASP.NET. Ejemplos de ello son los Web Forms y XML WebServices.



# Arquitectura (.NET)

- Proveer un entorno consistente de programación orientada a objetos donde el código objeto sea almacenado y ejecutado localmente, ejecutado localmente pero distribuido en Internet o ejecutado remotamente.
- Proveer un entorno de ejecución de código que minimice el deployment de los programas y resuelva los conflictos de versión.
- Proveer un entorno de ejecución de código que garantice la ejecución segura del código, incluyendo el código generado por desconocidos o terceras partes.
- Proveer un entorno de ejecución de código que elimine los problemas de performance en entornos con lenguaje batch o interpretado.
- Ofrecer una experiencia consistente al desarrollador sin importar si esta implementando aplicaciones para Windows o el Web.
- Construir toda la comunicación sobre estándares de la industria para asegurar que el código basado en .NET Framework sea integrable a otro código.



# Arquitectura (ASP.NET)

- **ASP.NET (1)**

- Es una implementación completamente nueva de ASP, escrita de cero en C#.
- ASP.NET utiliza lenguajes de programación compilados como Visual Basic.Net, C#, incluso COBOL (es “language-neutral”), para escribir aplicaciones Web.
- Las aplicaciones son compiladas en el servidor, y las páginas son generadas en HTML específicamente para el browser que hizo la invocación.



# Arquitectura (ASP.NET)

- **ASP.NET (2)**
  - Es un lenguaje compilado común que se ejecuta en el servidor.
  - Aplica conceptos de “early binding”, compilación “just-in-time”, optimización de código nativa y “caching services”.
  - Tiene mejor performance que ASP.
  - Tiene un conjunto de herramientas completo y un IDE común para diseño (VisualStudio.Net).
  - La .NET Framework class library, la mensajería, y las soluciones de acceso a datos son accesibles completamente desde el Web en forma transparente.
  - Es independiente al lenguaje, ya que permite elegir el lenguaje que más se aplique al problema o particionar el mismo e implementar la solución con múltiples lenguajes.

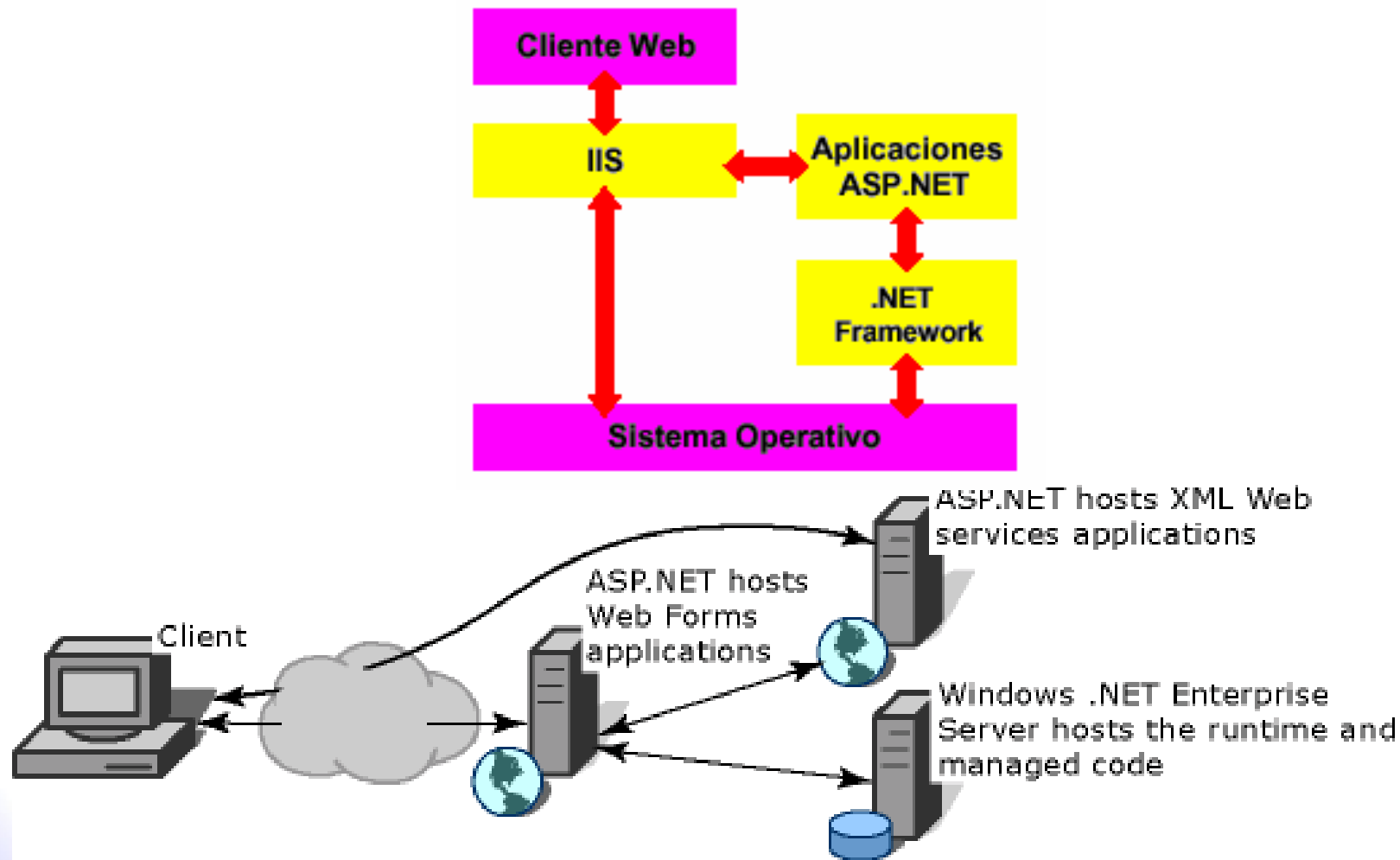


# Arquitectura (ASP.NET)

- **ASP.NET (3)**
  - Emplea una configuración a nivel de archivos de texto a nivel jerárquico que simplifica aplicar los “set” al entorno del servidor y las aplicaciones Web.
  - El deployment de una aplicación ASP.NET implica simplemente copiar los archivos necesarios al servidor.
  - Se integra en la autenticación del sistema operativo Windows y permite una configuración a nivel de aplicación.



# Arquitectura (ASP.NET)



01/04/2004

ASP.NET

17



# Arquitectura (ASP.NET)

- **Conceptos centrales de ASP.NET**
  - Separar presentación de lógica del negocio
  - Usar servicios provistos por el .NET Framework
  - El código es compilado la primera vez que se accede a una página
  - Administración de estado
  - Utilización de cualquier lenguaje
  - Actualizar archivos mientras se está ejecutando la aplicación



## Arquitectura (ASP.NET) Ejecución

- **ASP. NET es parte del entorno .NET (. NET framework ):**
  - La ejecución utiliza el soporte del CLR (Common Language Runtime ).
  - El Código en Segundo plano se compila al código intermedio de MS (MS-IL).
  - ASP. NET utiliza las ventajas de la plataforma multilenguaje, con la Biblioteca de clases común (BCL) y el sistema de tipos comunes (CTS).

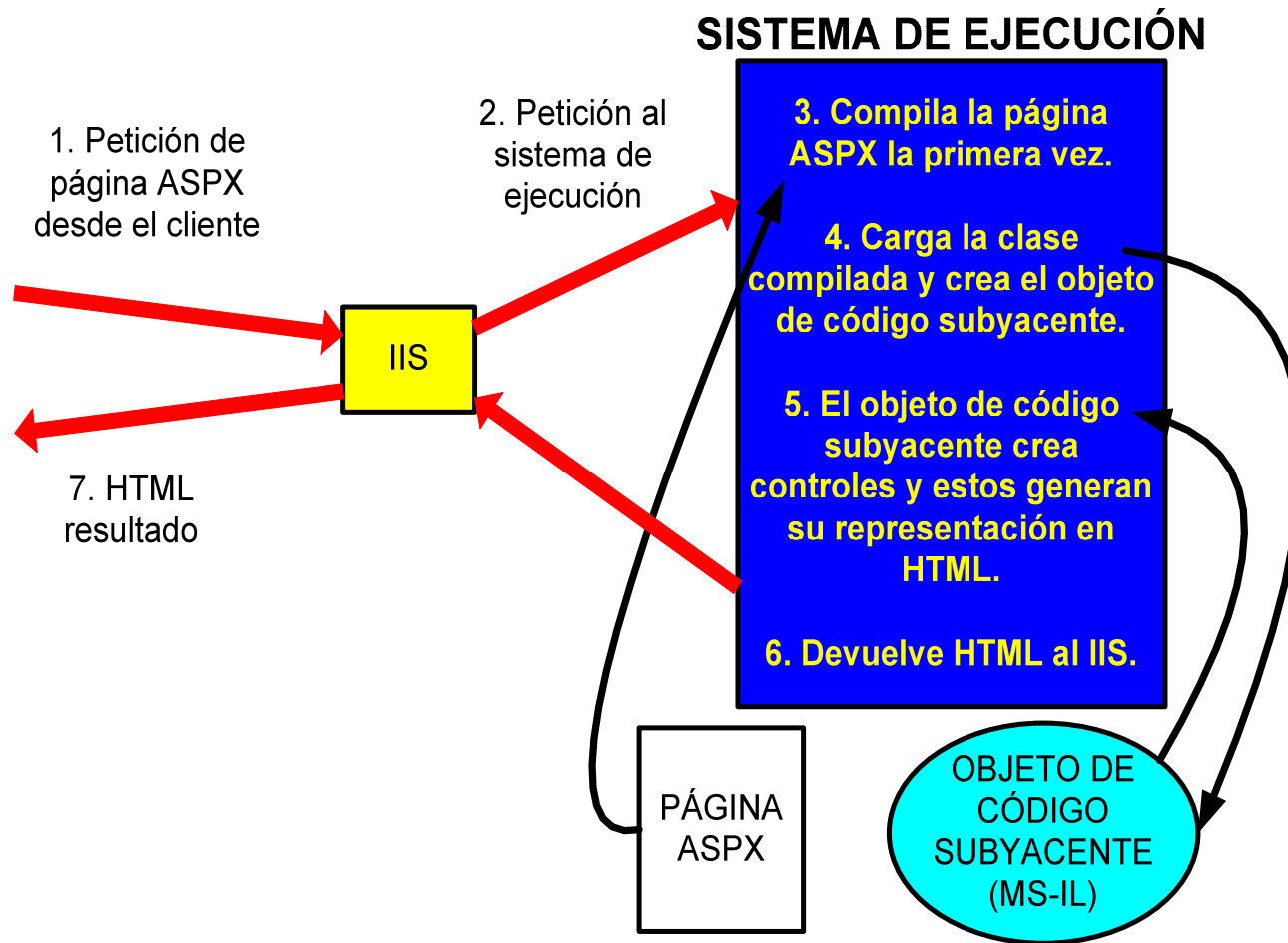


# Arquitectura (ASP.NET) Ejecución

- **HTTP Runtime**
  - Código administrado
    - Se ejecuta en un proceso no administrado
  - Permite 100% de disponibilidad
    - Procesa asincrónicamente todas las llamadas
    - Multithreaded
  - Reemplaza ISAPI
    - Internet Server Application Programming Interface



# Arquitectura (ASP.NET) Ejecución





## Arquitectura (ASP.NET) Tipos de Ficheros

- **Diferentes archivos, distinguibles por su extensión**
  - Archivos ASP.NET estándar:
    - .aspx o .ascx
  - Servicios Web :
    - .asmx
  - Archivos de código:
    - .cs, .vb, ...
  - Configuración:
    - Config.web
  - Aplicaciones Web :
    - Global.asax
- **Son todos Archivos de texto**
- **La forma más rápida de comenzar:**
  - Cambiar la extensión .asp por .aspx



## Arquitectura (ASP.NET) Sintaxis de la página

- **Directivas**

- `<%@ Page language="VB"%>`

- **Bloques de declaración de código**

- `<script runat="server" [language = ...]>`  
`[ líneas de código ] </script>`

- **Código de conversión (Render)**

- `<% [código en línea o expresión] %>`

- **Sintaxis de controles HTML**

- `<HTMLtag runat="server" [attribute = ...]> </HTMLtag>`



## Arquitectura (ASP.NET) Sintaxis de la página

- **Expresión de vinculación a datos**
  - `<%# Expresión de vinculación %>`
- **Marcadores de objetos del lado del servidor**
  - `<object id="id" runat="server" identifier="Nombre">`
- **Directivas de inclusión en el servidor**
  - `<!-- #include Tipo = Archivo -->`
- **Comentarios en el servidor**
  - `<%-- Comentario --%>`



## Arquitectura (ASP.NET) Sintaxis de los controles

- **Controles del Lado del servidor**
  - `<ASP:TextBox id="MyTb1" runat="server">`
- **Propiedades del control del servidor**
  - `<ASP:TextBox maxlength="80" runat="server">`
- **Sub propiedad (del lado del cliente)**
  - `<ASP:Label font-size="14" runat="server">`
- **Vinculación a eventos del control**
  - `<ASP:Button OnClick="MyClick" runat="server">`



# Marco de Páginas

- **El marco de páginas de ASP.NET es un modelo de programación escalable que puede usar en el servidor para generar páginas Web dinámicamente**
- **Visual Studio .NET provee un nuevo paquete de “Windows-based forms”.**
  - Todos los lenguajes que corren sobre .NET comparte el mismo diseñador de forms.
  - Además los Windows Forms que uno crea en Visual Studio .NET pueden ser heredados a otros proyectos implementados en otros lenguajes.
  - Se consolidaron los controles que se utilizan en los Windows Forms y los Web Forms permitiendo así tener una interface común sin importar el tipo de aplicación.
  - Esto hace posible que desarrolladores puedan migrar fácilmente de un lenguaje a otro.



# Controles de servidor

- La mayoría de los controles Microsoft® ASP.NET se han diseñado para que no dependan del explorador; generan un formato HTML sencillo que la mayoría de los exploradores pueden reconocer y procesar con facilidad.
- Como desarrollador de páginas, puede controlar hasta cierto punto la versión del lenguaje HTML que se utiliza para generar la respuesta del explorador.
- Algunos de los controles ASP.NET (por ejemplo, los controles de servidor de validación) detectan las capacidades del explorador subyacente y adaptan su respuesta en consecuencia.
- En muchos casos, esto significa que el formato se enriquece con código de archivo de comandos que mejora la funcionalidad global del control.



## Controles de servidor

- Como su propio nombre indica, los controles de servidor ASP.NET se diseñan y se programan como componentes del servidor.
- Los controles de servidor ASP.NET son clases administradas, pertenecen a Microsoft® .NET Framework y siguen el paradigma orientado a objetos. Su salida es código de formato que un explorador puede consumir.
- En realidad pertenecen al servidor pero se representan en HTML mediante diferentes mecanismos, entre ellos javascript.



## Controles de servidor

- **Sintaxis de los controles ASP.NET**
  - `<asp:Nombre Atributos />`
  - Nombre
    - TextBox, DropDownList, etc.
  - Atributos
    - `Id=IDdelControl`
    - `runat=server`

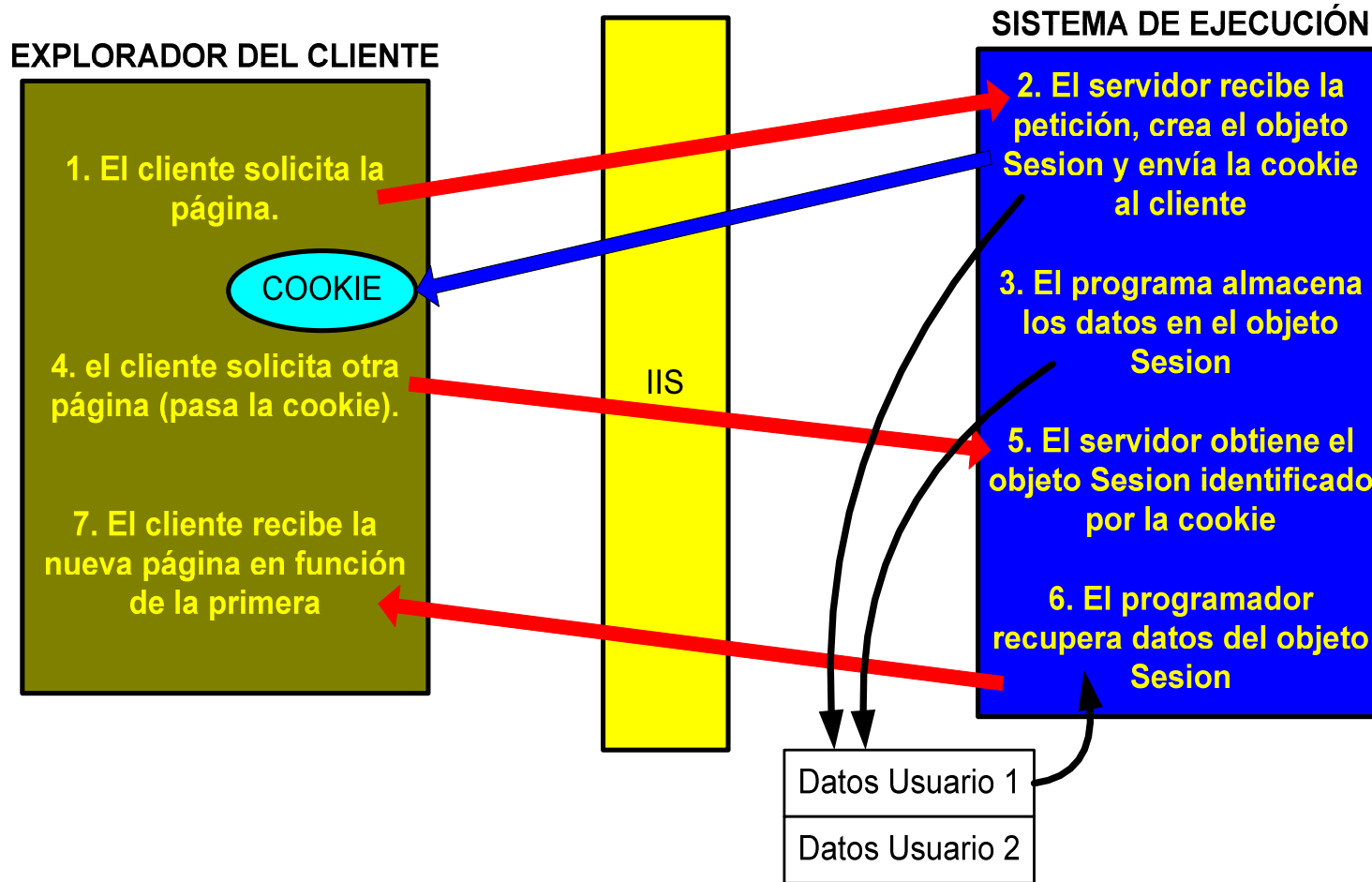


# Administración de estado

- **¿Que es una sesión?**
  - Restringida a una aplicación lógica
  - Contexto en el cual un cliente se conecta con un servidor
- **Funcionalidad**
  - Solicitud de identificación y calificación
  - Almacenar Datos entre llamadas
  - Eventos de Sesión
  - Liberación de los datos de Sesión
- **Proceso de estado en el Servidor .NET**



# Administración de estado





## Almacenamiento en caché

- **Mejora el rendimiento de la aplicación Web**
- **Caché de salida**
  - Almacena y obtiene páginas y objetos
  - Caché de página
  - Caché de fragmentos
- **Reglas de expiración**
- **APIs**
  - Permite personalizar las acciones del Caché



## Enlace de datos

- **ASP.NET presenta una nueva sintaxis declarativa, `<%# %>`. Esta sintaxis es la base para utilizar el enlace de datos en una página .aspx.**
- **Todas las expresiones de enlace de datos deben estar contenidas dentro de estos caracteres. En los ejemplos siguientes se ilustra el enlace de datos simple de varios orígenes:**
  - Propiedad Simple => *Cliente*:
    - `<%# custID %>`
  - Colección => *Pedidos*:
    - `<asp:ListBox id="List1" datasource='<%# myArray %>' runat="server">`
  - Expresión => *Contacto*:
    - `<%# ( customer.First Name + " " + customer.LastName ) %>`
  - Resultado del método => *Saldo pendiente*:
    - `<%# GetBalance(custID) %>`



# Seguridad

- **Motivos**
  - Prevenir el acceso a áreas del Servidor Web
  - Registrar y almacenar información relevante de los usuarios
- **Configuración de Seguridad**
  - Tag <Security> en el archivo Config.web
- **Autenticación, Autorización, Impersonalización**
- **Seguridad de acceso al código**
  - ¿es éste realmente el código original del servidor?
  - Proteger el servidor de “código malicioso”



# Seguridad

- **Autenticación**
  - Validar credenciales del usuario
  - Utilizar identidades de autenticación
  - Tipos de Autenticación
    - Windows, integrada con IE 5.0
    - Passport, servicios centralizados provistos por Microsoft
    - Cookie, adjunto en el requerimiento
- **Autorización**
  - Determinar cuando es permitido un requerimiento
  - Autorización por Archivo y por URL



# Seguridad

- **Impersonalización**
  - IE autentica al “usuario”
  - Se pasa un “token” a la aplicación ASP.NET
  - ASP.NET lo impersona
  - El acceso se permite de acuerdo a las asignaciones por NTFS
- **Seguridad de Acceso al código**
  - Característica del .NET Framework
  - Verifica la identidad del código y su origen
  - Especifica las operaciones que el código tiene permitido ejecutar



# Seguridad

- **Autenticación de Windows**
  - Delega el proceso de autenticación en el IIS.
  - El IIS muestra un cuadro de diálogo en el usuario introduce su identificación y contraseña de Windows.
  - Adecuada para una intranet de Windows, cuando sabemos a priori el dominio de usuario que tiene la aplicación.



# Seguridad

- **Autenticación basada en formularios**
  - La aplicación web proporciona al usuario una forma de registrarse.
  - Para entrar en la parte privada del sitio se pide al usuario que introduzca usuario y contraseña.
  - Permite la entrada si coinciden con los registrados.
  - Se identifica al usuario autenticado mediante una cookie.
  - Este tipo de autenticación se configura mediante una entrada en el fichero web. config.



# Seguridad

- **Autenticación de Passport (I)**
  - Problema de la autenticación a través de formularios: cada sitio web tiene su usuario y contraseña y hay que recordarlas todas.
  - Con .net Passport el usuario se registra en Passport y cuando las aplicaciones web necesitan autenticar a un usuario utilizan passport como intermediario.
  - En realidad el usuario sólo tiene que recordar su usuario y contraseña en Passport.



# Seguridad

- **Autenticación de Passport (II)**

- Se puede almacenar información asociada a un usuario: dirección, nº tarjeta.
- Así se pueden automatizar la acción de rellenar formularios de entrega.
- Cómo establecer este tipo de autenticación:
  - Firmar la licencia con Microsoft para utilizar Passport.
  - Descargar y utilizar el SDK de Passport.
  - Establecer el fichero web.config.
  - Escribir el código correspondiente.



# Seguridad

- **Autorización**

- Según el usuario que sea decidir si se debe. permitir que vea la página que solicita.
- En ASP. NET se puede limitar de forma declarativa el acceso a página en el fichero web.config.
  - Sección: <authorization>.
  - Elementos: <allow> y <deny>.
- Se puede especificar por usuario o por grupo.
  - ? => usuarios an usuarios anónimos.
  - \* => todos los usuarios.



# Seguridad

- **El problema de la identidad**
  - Las páginas de un sitio web en las que controlamos el acceso representan un nivel intermedio en un sistema de tres niveles.
  - El tercer nivel es el nivel de datos que normalmente posee sus propios mecanismos de seguridad.
  - Para poder realizar las operaciones dependemos de lo que el nivel de datos considere que es la identidad del usuario que realiza la petición de servicios.



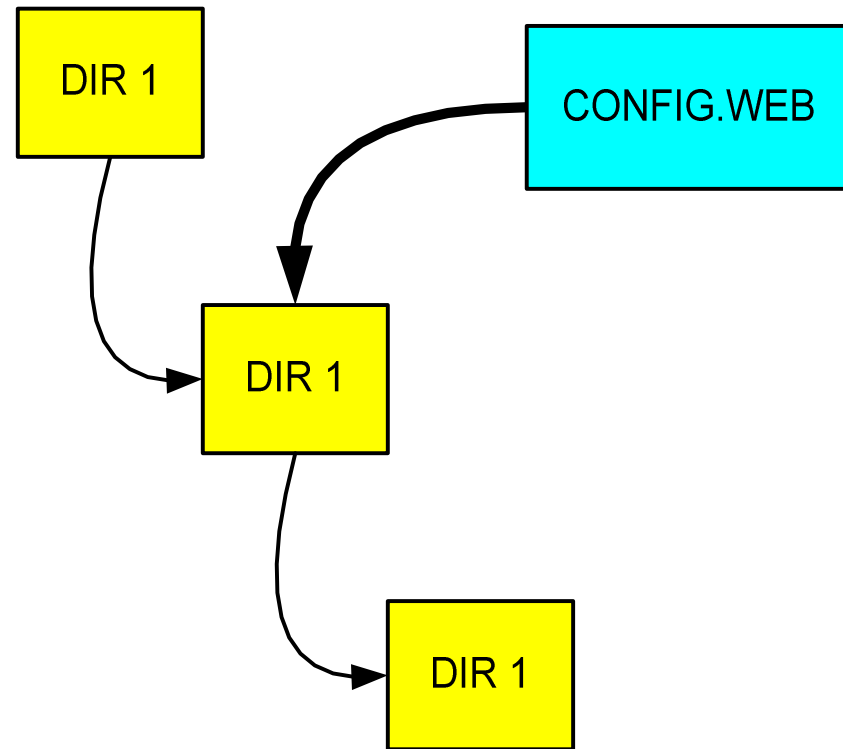
# Seguridad

- **Identidad en ASP. NET**
  - Modelo de **usuario de confianza**. Se asigna una identidad para acceder al nivel de datos que permita realizar determinadas operaciones.
  - Elemento **<identity>** del fichero web.config.
  - Modelo **delegación de personalidad**. El código de la página adopta la identidad del usuario autenticado.
    - Es necesario hacer dos autenticaciones.
  - Elemento **<impersonation>** de web.config.



# Configuración

- **Conceptos y Arquitectura**
  - Arch. de configuración: Config.web
    - Basado en XML, legible y modificable por “humanos”
    - El archivo se mantiene en el mismo directorio que la aplicación
    - Los cambios se detectan automáticamente
  - Arquitectura de configuración jerárquica
    - Afecta el subdirectorio actual y todos los dependientes





# Configuración

```
<configuration>

    <configsections>

        <add names="httpmodules"
            type="System.Web.Config.httpModulesConfigHandler"/>

        <add names="sessionstate" type="..."/>

    </configsections>

    <httpmodules>
        <!-- Subelementos de http -->
    </httpmodules>

    <sessionstate>
        <!-- Subelementos de estado de sesión -->
    </sessionstate>

</configuration>
```



# Conclusiones

- **Ventajas de .NET**
  - Sistema integrado (funciona como un todo)
  - Posee la misma potencialidad que cualquier otro sistema, pero está adaptado a la mayor plataforma (MS-Windows)
  - Facilidad de desarrollo (¿Puede ser una desventaja?)
  - Sistema de seguridad integrado
  - Compatible con las versiones anteriores ASP



# Conclusiones

- **Desventajas de .NET (Acusaciones)**
  - De ser conjunto de productos en lugar de un conjunto de estándares.
  - De ser el resultado de una estrategia de producto.
  - De ser una reescritura de Windows DNA (mejorando sus propias tecnologías como MTs, COM+, Message Queue Server y MS-SQL).
  - De tratar de imponer C#. Dice que .NET es a C#, como J2EE a Java.



# Referencias

- Microsoft MSDN <http://www.microsoft.com/spanish/msdn>
- Tutoriales de Microsoft  
<http://es.gotdotnet.com/quickstart/default.aspx>
- Planeta ASP.NET (Comunidad de desarrolladores)  
<http://www.planetaasp.net/>
- Libros sobre ASP.NET  
<http://www.microsoft.com/spanish/msdn/comunidad/comunidades/asp-aspnet/libros/default.asp>